

# Numblex 0.2.0 Manual

## Concepts

### Numbering

Numbering is just a function that takes a list of numbers and returns a string. And this function can be used as the value of the numbering option in Typst.

For example, the numbering function could return a string like this:

<1-(3).4.>

We can split the string into several parts, and each part is an element.

<1-(3).4.>

### Element

The elements are categorized into two types: ordinals and constants. Here “<”, “-”, “>” are constants, and “1”, “(3)”, “4.” are ordinals.

We use the following format to represent the whole numbering

### Numbering String

The Typst official numbering string is not powerful enough, we usually need to set the numbering to a custom function to implement more complex numbering. However, this leads to redundancy. We define a new format to represent the numbering.

{<} {[1]} {-} {[3]}. {[4]}. {>}

Each element is enclosed in a pair of curly braces({}), and anything else is ignored. The element can be a constant(with no [] in it) or an ordinal(with [] in it). The ordinal element is only displayed when the depth is enough.

### Patterns

Patterns are used to represent the ordinal or constant. The ordinal will be replaced by the final ordinal string in the output numbering, and anything outside the [] will be kept as it is.

{Chapter [1]}. ⇒ Chapter 1.

Of course, this is designed to avoid the following problem:

```
#set heading(numbering: "Chapter 1.")  
= Once Upon a Time  
#set heading(numbering: "Ch\apter 1.")  
= Once Upon a Time
```

**Cha. Once Upon a Time**

**Ch\b. Once Upon a Time**

### Ordinals

Most of the time, you can just put the character corresponding to the ordinal you want in the []. numblex passes the character to the numbering function in Typst to get the ordinal.

However, numblex has also modified and extended the ordinal definition.

- `[]`: an empty string, it takes the number but generates nothing.
- `[(1)]`: shorthand for circled numbers (①, ②, ③, ...). If you want the ordinal (1), (2), (3), ..., please use `{[(1)]}` instead.

Use single character to represent the ordinal if possible, since Typst have complicated rules to handle the prefix and suffix of the numbering.

The element can be different in different contexts. For example, we might need them to show up in different forms according to the depth.

## Conditions

Conditions are functions that take the current numbering configuration and return a boolean value. The condition match is executed sequentially, and the first match will be used.

Currently the following configuration context are defined:

### depth: int (short: d)

The depth of the numbering.

From Typst v0.11.1 on, we can use `heading.depth` to get the depth of the heading. Similarly, we introduce the numbering depth, which is the length of the list of numbers passed to the numbering function.

### ⚠ Warning

The condition here is implemented using `eval`, which is not safe and might cause other problems.

You may represent a conditional element using the following format:

$$\{\text{PAT\_1:COND\_1;PAT\_2:COND\_2;...}\}$$

Leave the condition empty is equivalent to `true`.

$$\{\text{PAT\_1:COND\_1;PAT\_2}\} \equiv \{\text{PATTERN\_1:COND\_1;PAT\_2:true}\}$$

If no condition is matched, the element will return an empty string. Notice that `[]` must appear in *none(constant element)* or *all(ordinal element)* of the patterns.

## Examples

```
#set heading(numbering: numblex("{Section
[A].:d==1;[A].}{[1].}{[1]}}"))
= Electrostatic Field
== Introduction
=== Gauss's Law
```

## Section C. Electrostatic Field

### C.1. Introduction

#### C.1.1) Gauss's Law

```
#set heading(numbering:
numblex("{[-].:d==1}{[1].:d==2}
{[(1)]}"))
= 保角变换
== 介绍
=== 调和函数的保角性
```

## 四、保角变换

### 1. 介绍

#### ① 调和函数的保角性

```
#let example = numblex("{<} {[1]}  
{-:d>=2} {[1].} {[1].} {>}")  
#numbering(example, 1) \  
#numbering(example, 1, 3) \  
#numbering(example, 1, 3, 4)
```

```
<1>  
<1-(3).>  
<1-(3).4.>
```

## Reference

The package provides a function `numblex` to generate the numbering function for Typst.

### function `numblex`

#### Parameters

- `positional` `s`
- `named` `..options`

## Known Issues & Limitations

1. Automatic repeating (which Typst official numbering supports) is not supported yet.
2. Character escaping for “`{[]:;}`” is not supported yet.